

---

# Fitting community models to large sparse networks

---

**Aiyou Chen**

Google, Inc  
1600 Amphitheatre Pkwy  
Mountain View, CA 94043  
aiyouchen@google.com

**Arash A. Amini**

Department of Statistics  
University of Michigan  
Ann Arbor, MI 48109-1107  
aaamini@umich.edu

**Peter J. Bickel**

Department of Statistics  
University of California, Berkeley  
Berkeley, CA 94760  
bickel@stat.berkeley.edu

**Elizaveta Levina**

Department of Statistics  
University of Michigan  
Ann Arbor, MI 48109-1107  
elevina@umich.edu

## Abstract

Many algorithms have been proposed for fitting network models with communities but most of them do not scale well to large networks, and often fail on sparse networks. Here we propose a new fast pseudo-likelihood method for fitting the stochastic block model for networks, as well as a variant that allows for an arbitrary degree distribution by conditioning on degrees. We show that the algorithms perform well under a range of settings, including on very sparse networks, and illustrate on the example of a network of political blogs. We also propose spectral clustering with perturbations, a method of independent interest, which works well on sparse networks where regular spectral clustering fails, and use it to provide an initial value for pseudo-likelihood.

## 1 Introduction

Analysis of network data is important in a range of disciplines and applications, appearing in such diverse areas as sociology, epidemiology, computer science, and national security, to name a few. Network data here refers to observed edges between nodes, possibly accompanied by additional information on the nodes and/or the edges, e.g., edge weights. One of the fundamental questions in analysis of such data is detecting and modeling community structure within the network. A lot of algorithmic approaches to community detection have been proposed, mainly in the physics literature (see [1], [2] for reviews). These include various greedy methods such as hierarchical clustering (see [3] for a review) and algorithms based on optimizing a global criterion over all possible partitions, such as normalized cuts [4] and modularity [5]. The statistical learning community has been more focused on model-based methods, which postulate and fit a probabilistic model for a network with communities. These include the popular stochastic block model [6], its extensions to include varying degree distributions within communities [7] and overlapping communities [8, 9], and various latent variable models [10, 11].

The stochastic block model is perhaps the most commonly used and best studied model for community detection. For a network with  $n$  nodes defined by its  $n \times n$  adjacency matrix  $A$ , this model postulates that the true node labels  $c = (c_1, \dots, c_n) \in \{1, \dots, K\}^n$  are drawn independently from the multinomial distribution with parameter  $\pi = (\pi_1, \dots, \pi_K)$ , where  $K$  is the number of communities, assumed known. Conditional on the labels, the edge variables  $A_{ij}$  for  $i < j$  are independent Bernoulli variables with

$$\mathbb{E}[A_{ij} | c] = P_{c_i c_j}, \quad (1)$$

where  $P = [P_{ab}]$  is a  $K \times K$  symmetric matrix. The network is undirected, so  $A_{ji} = A_{ij}$ , and  $A_{ii} = 0$  (no self-loops). The problem of community detection is then to infer the node labels  $c$  from  $A$ , which typically also involves estimating  $\pi$  and  $P$ .

There are many extensions of the block model but we will only mention one here that we will use later in the paper. The block model implies the same expected degree for all nodes within a community, which excludes networks with “hub” nodes commonly encountered in practice. The degree-corrected block model [7] removes this constraint by replacing (1) with  $\mathbb{E}[A_{ij}|c] = \theta_i \theta_j P_{c_i c_j}$ , where  $\theta_i$ ’s are node degree parameters which satisfy an identifiability constraint. If the degree parameters only take on a discrete number of values, one can think of the degree-corrected block model as a regular block model with a larger number of blocks, but that loses the original interpretation of communities. In [7] the Bernoulli distribution for  $A_{ij}$  was replaced by the Poisson, primarily for ease of technical derivations, and in fact this is a good approximation for a range of networks [12].

Fitting block models is non-trivial, especially for large networks, since in principle the problem of optimizing over all possible label assignments is NP-hard. In the Bayesian framework, Markov Chain Monte Carlo methods have been developed [13, 14] but they only work for networks with a few hundred nodes. Variational methods have also been developed and studied (see for example [8, 15, 16]), and are generally substantially faster than the Gibbs sampling involved in MCMC, but still do not scale to the order of a million nodes. A profile likelihood approach was proposed in [17]: since for a given label assignment parameters can be estimated trivially by plug-in, they can be profiled out and the resulting criterion can be maximized over all label assignments by greedy search. The same method is used in [7] to fit the degree-corrected block model. The speed of these algorithms depends on exactly what search method is used and the number of iterations it is run for, but again these generally work well for thousands but not millions of nodes. A method of moments approach was proposed in [18], for a large class of network models that includes the block model as a special case. The generality of this method is an advantage, but it involves counting all occurrences of specific patterns in the graph, which is computationally challenging beyond simple special cases. Some faster approximations for block model fitting based on spectral representations are also available [19, 20], but the properties of these approximations are only partially known.

In this paper, we propose a new fast pseudo-likelihood algorithm for fitting the block model, as well as a variation conditional on node degrees that allows for fitting networks with highly variable node degrees within communities. The idea of pseudo-likelihood dates back to [21], and in general amounts to ignoring some of the dependency structure of the data in order to simplify the likelihood and make it more tractable. The main feature of the adjacency matrix we ignore here is its symmetry; we also apply block compression, that is, divide the nodes into blocks and only look at the likelihood of the row sums within blocks. This leads to an accurate and fast approximation to the block model likelihood, which allows us to easily fit networks with tens of millions of nodes. We also propose a regularized version of spectral clustering, a new clustering method of independent interest which we use to initialize pseudo-likelihood. For sparse networks, regular spectral clustering often performs very poorly, likely due to the presence of many disconnected components. We perturb the network by adding additional weak edges to connect these components, resulting in regularized spectral clustering which performs well under a wide range of settings. In the rest of the paper, we lay out the algorithms, and demonstrate their performance on a range of simulated networks as well as on a network of political blogs. The theoretical properties of the algorithms are postponed to a later journal paper.

## 2 Algorithms

### 2.1 Pseudo-likelihood

The joint likelihood of  $A$  and  $c$  could in principle be maximized via the expectation-maximization (EM) algorithm, but the E-step involves optimizing over all possible label assignments, which is in principle NP-hard. Instead, we introduce an initial labeling vector  $e = (e_1, \dots, e_n)$  which partitions the nodes into  $K$  groups (each  $e_i$  takes values in  $\{1, \dots, K\}$ ). Note that for convenience we partition into the same number of groups as we assume to exist in the true model, but in principle the same idea can be applied with a different number of groups; in fact dividing the nodes into  $n$  groups with a single node in each group instead gives an algorithm equivalent to that of [22].

The main quantity we work with are the block sums along the columns

$$b_{ik} = \sum_j A_{ij} 1(e_j = k) \quad (2)$$

for  $i = 1, \dots, n, k = 1, \dots, K$ . Let  $\mathbf{b}_i = (b_{i1}, \dots, b_{iK})$ . Further, let  $R$  be the  $K \times K$  matrix with entries  $\{R_{ka}\}$  given by

$$R_{ka} = \frac{1}{n} \sum_{i=1}^n 1(e_i = k, c_i = a). \quad (3)$$

Let  $R_{k\bullet}$  be the  $k$ th row of  $R$ , and let  $P_{\bullet l}$  be the  $l$ th column of  $P$ . Let  $\lambda_{lk} = nR_{k\bullet}P_{\bullet l}$  and  $\Lambda = \{\lambda_{lk}\}$ .

Our approach is based on the following key observations: for each node  $i$ , conditional on labels  $c = (c_1, \dots, c_n)$  with  $c_i = l$ ,

(A)  $\{b_{i1}, \dots, b_{iK}\}$  are mutually independent;

(B)  $b_{ik}$ , a sum of independent Bernoulli variables, is approximately Poisson with mean  $\lambda_{lk}$ .

With true labels  $\{c_i\}$  unknown, each  $\mathbf{b}_i$  can be viewed as a mixture of Poisson vectors. A necessary condition for the identifiability of parameters  $(\pi, RP)$ , or equivalently  $(\pi, \Lambda)$ , is that the means of the Poisson mixture components are different, i.e.,  $\Lambda$  has no two identical rows. This has implications for the theoretical analysis of the method, but we do not focus on this here.

By ignoring the dependence among  $\{\mathbf{b}_i, i = 1, \dots, n\}$ , treating  $\{c_i\}$  as latent variables, and setting  $\lambda_l = \sum_k \lambda_{lk}$ , we can write the pseudo log-likelihood as follows (up to a constant):

$$\ell_{\text{PL}}(\pi, \Lambda; \{\mathbf{b}_i\}) = \sum_{i=1}^n \log \left( \sum_{l=1}^K \pi_l e^{-\lambda_l} \prod_{k=1}^K \lambda_{lk}^{b_{ik}} \right) \quad (4)$$

A pseudo-likelihood estimate of  $(\pi, \Lambda)$  can then be obtained by maximizing  $\ell_{\text{PL}}(\pi, \Lambda; \{\mathbf{b}_i\})$ . This can be done via the EM algorithm for mixture models, a fairly standard tool, which alternates updating parameter values with updating probabilities of node labels. Once the EM converges, we update the initial partition vector  $e$  to the most likely label for each node as indicated by EM, and repeat this process for a fixed number of iterations  $T$ .

For any labeling  $e$ , we use the following notation:  $n_k(e) = \sum_i 1(e_i = k)$ ,  $n_{kl}(e) = n_k(e)n_l(e)$  if  $k \neq l$ ,  $n_{kk}(e) = n_k(e)(n_k(e) - 1)$ , and  $O_{kl}(e) = \sum_{i,j} A_{ij} 1(e_i = k, e_j = l)$ . We will suppress the dependence on  $e$  whenever there is no ambiguity.

**The pseudo-likelihood algorithm.** Initialize labels  $e$ , and let  $\hat{\pi}_l = n_l/n$ ,  $\hat{R} = \text{diag}(\hat{\pi}_1, \dots, \hat{\pi}_K)$ ,  $\hat{P}_{lk} = O_{lk}/n_{lk}$ ,  $\hat{\lambda}_{lk} = n\hat{R}_{k\bullet}\hat{P}_{\bullet l}$ ,  $\hat{P} = \{\hat{P}_{lk}\}$  and  $\hat{\Lambda} = \{\hat{\lambda}_{lk}\}$ . Then repeat  $T$  times:

1. Compute the block sums  $\{b_{il}\}$  according to (2),
2. Using current parameter estimates  $\hat{\pi}$  and  $\hat{\Lambda}$ , estimate probabilities for node labels by

$$\hat{\pi}_{il} = \mathbb{P}_{\text{PL}}(c_i = l | \mathbf{b}_i) = \frac{\hat{\pi}_l \prod_{m=1}^K \exp(b_{im} \log \hat{\lambda}_{lm} - \hat{\lambda}_{lm})}{\sum_{k=1}^K \hat{\pi}_k \prod_{m=1}^K \exp(b_{im} \log \hat{\lambda}_{km} - \hat{\lambda}_{km})}.$$

3. Given label probabilities, update parameter values as follows:

$$\hat{\pi}_l = \frac{1}{n} \sum_{i=1}^n \hat{\pi}_{il}, \quad \hat{\lambda}_{lk} = \frac{\sum_i \hat{\pi}_{il} b_{ik}}{\sum_i \hat{\pi}_{il}}.$$

4. Return to step 2 unless the parameter estimates have converged.
5. Update labels by  $e_i = \arg \max_l \hat{\pi}_{il}$  and return to step 1.
6. Update  $\hat{P}$  as follows:  $\hat{P}_{lk} = (\sum_{i,j} A_{ij} \hat{\pi}_{il} \hat{\pi}_{jk}) / n_{lk}(e)$ .

In practice, in step 6 we only include the terms corresponding to  $\hat{\pi}_{il}$  greater than some small threshold. The EM method is fitting a valid mixture model and is thus guaranteed to converge to a stationary point of the objective function [23]. Another option is to update labels after every parameter update (that is, skip step 4.) We have found empirically that the algorithm above is more stable, and converges faster. In general, we only need a few label updates until convergence, and even using  $T = 1$  (one-step label update) gives reasonable results with a good initial value. The choice of the initial value of  $e$ , on the other hand, can be important; see more on this in Section 2.3.

## 2.2 Pseudo-likelihood conditional on node degrees

For networks with hub nodes or those with substantial degree variability within communities, the block model can provide a poor fit, essentially dividing the nodes into low-degree and high-degree groups [7, 24]. The extension of the block model designed to cope with this situation, the degree-corrected block model [7], has an extra degree parameter to be estimated for every node, and writing out a pseudo-likelihood that lends itself to an EM-type optimization is more complicated. However, there is a simple alternative: consider the pseudo-likelihood conditional on the observed node degrees. Whether these degrees are similar or not will not then matter, and the fitted parameters will reflect the underlying block structure rather than the similarities in degrees.

The conditional pseudo-likelihood is again based on a simple observation:

- (C) If random variables  $X_k$  are independent Poisson with means  $\mu_k$ , their distribution conditional on  $\sum_k X_k$  is multinomial.

Applying this observation to the variables  $(b_{i1}, \dots, b_{iK})$ , we have that their distribution conditional on labels  $c$  with  $c_i = l$  and the node degree  $d_i = \sum_k b_{ik}$  is multinomial with parameters  $\theta_{lk} = \frac{\lambda_{lk}}{\lambda_l}$ . The conditional log pseudo-likelihood (up to a constant) is then given by,

$$\ell_{\text{CPL}}(\pi, \Theta; \{\mathbf{b}_i\}) = \sum_{i=1}^n \log \left( \sum_{l=1}^K \pi_l \prod_{k=1}^K \theta_{lk}^{b_{ik}} \right) \quad (5)$$

and the parameters can be obtained by maximizing this function via the EM algorithm for mixture models, as before. We again repeat the EM for a fixed number of iterations updating the initial partition vector after the EM has converged. The algorithm is then the same as that for unconditional pseudo-likelihood, with steps 2 and 3 replaced by:

- 2'. Based on current estimates  $\hat{\pi}$  and  $\{\hat{\theta}_{lk}\}$ , let

$$\hat{\pi}_{il} = \mathbb{P}_{\text{CPL}}(c_i = l | \mathbf{b}_i) = \frac{\hat{\pi}_l \prod_{m=1}^K \hat{\theta}_{lm}^{b_{im}}}{\sum_{k=1}^K \hat{\pi}_k \prod_{m=1}^K \hat{\theta}_{km}^{b_{im}}}.$$

- 3'. Given label probabilities, update parameter values as follows:

$$\hat{\pi}_l = \frac{1}{n} \sum_{i=1}^n \hat{\pi}_{il}, \quad \hat{\theta}_{lk} = \frac{\sum_i \hat{\pi}_{il} b_{ik}}{\sum_i \hat{\pi}_{il} d_i}.$$

## 2.3 Initializing the partition vector

We now turn to the question of how to initialize the partition vector  $e$ . Note that the full likelihood, pseudo-likelihoods  $\ell_{\text{PL}}$  and  $\ell_{\text{CPL}}$ , and other standard objective functions such as modularity can all be multi-modal. The numerical results in Section 3 suggest that the initial value cannot be entirely arbitrary, but the results are not too sensitive to it. We will quantify this further in Section 3; here we describe the two options we use as initial values, both of which are of independent interest as clustering algorithms for networks.

### 2.3.1 Clustering based on 1- and 2-degrees

One of the simplest possible ways to group nodes in a network is to separate them by degree, say by one-dimensional  $K$ -means clustering applied to the degrees as in [25]. This only works for certain

types of block models identifiable from their degree distributions, and in general  $K$ -means does not deal well with data with many ties, which is the case with degrees. Instead, we consider two-dimensional  $K$ -means clustering on the pairs  $(d_i, d_i^{(2)})$ , where  $d_i^{(2)}$  is the number of paths of length 2 from node  $i$ , which can be obtained by summing the rows of  $A^2$ .

### 2.3.2 Spectral clustering with perturbations

A more sophisticated clustering scheme is based on spectral properties of the adjacency matrix  $A = \{A_{ij}\}$  or its graph Laplacian. Let  $D = \text{diag}(d_1, \dots, d_n)$  be diagonal matrix collecting node degrees. A common approach is to look at the eigenvectors of the normalized graph Laplacian  $L = D^{-1/2}AD^{-1/2}$ , choosing a small number, say  $r = K - 1$ , corresponding to  $r$  largest (in absolute value) eigenvalues, with the largest eigenvalue omitted (see, e.g., [4]). These vectors provide an  $r$ -dimensional representation for nodes of the graph, on which we can apply  $K$ -means to find clusters; this is one of the versions of spectral clustering, which was analyzed in the context of the block model by [20].

We found that this version of spectral clustering tends to do poorly at community detection when applied to sparse graphs, say, with expected degree  $\lambda < 5$ . The  $r$ -dimensional representation seems to collapse to a few points, likely due to the presence of many disconnected components. We have found, however, that a simple modification performs surprisingly well even for values of  $\lambda$  close to 1. The idea is to connect all disconnected components which belong to the same community by adding artificial “weak” links. To be precise, we “regularize” the adjacency matrix  $A$  by adding  $\alpha/p \times \lambda/n$  multiplied by the adjacency matrix of an Erdos-Renyi graph on  $n$  node with edge probability  $p$ , where  $\alpha$  is a constant. We found that, empirically,  $\alpha/p = 0.25$  works well for the range of  $n$  considered in our simulations, and that the results are essentially the same for all  $p > 0.1$ . Thus we make the simplest and computationally cheapest choice of  $p = 1$ , adding a constant matrix of small values to the original adjacency matrix. The rest of the steps, i.e., forming the Laplacian, obtaining the spectral representation and applying  $K$ -means, are performed on this regularized version of  $A$ . We will refer to this algorithm as spectral clustering with perturbations (SCP), as we perturb the network by adding new low-weight “edges”.

## 3 Numerical results

Here we investigate the performance of both the unconditional and conditional pseudo-likelihood algorithms on simulated networks, as well as that of spectral clustering with perturbations. We simulate two scenarios, one from the regular stochastic block model and one from the degree-corrected block model, to assess the performance in the presence of hub nodes. Throughout this section, we fix  $K = 3$  and  $\pi = (1/3, 1/3, 1/3)$ . Conditional on the labels, the edges are generated as independent Bernoulli variables with probabilities proportional to  $\theta_i \theta_j P_{ij}$ . The parameters  $\theta_j$  are drawn independently from the distribution of  $\Theta$  with  $\mathbb{P}(\Theta = 0.2) = \gamma$ ,  $\mathbb{P}(\Theta = 1) = 1 - \gamma$ . We do not enforce the identifiability scaling constraint on  $\theta$  at this point as it is absorbed into the scaling of the matrix  $P$  in (6) below. We consider two values of  $\gamma$ :  $\gamma = 0$ , which corresponds to the regular block model, and  $\gamma = 0.9$ , which corresponds to a network where 10% of the nodes can be viewed as hubs.

The matrix  $P$  is constructed as follows. It is controlled by two parameters: the “out-in-ratio”  $\beta$  [26], which we will vary from 0 to 0.2, and the weight vector  $w$ , which determines the relative degrees within communities. We consider two values of  $w$ :  $w = (1, 1, 1)$  (no information about communities is contained in node degrees) and  $w = (1, 5, 10)$  (degrees themselves provide relevant information for clustering). If  $\beta = 0$ , we set  $P^{(0)} = \text{diag}(w)$ , a diagonal matrix. Otherwise, we set the diagonal of  $P^{(0)}$  to  $\beta^{-1}w$  and set all off-diagonal elements to 1. We then fix the overall expected network degree  $\lambda$ , which is the natural parameter to control [17] and which we will vary from 1 to 15. Then we rescale  $P^{(0)}$  to obtain this expected degree, giving the final  $P$

$$P = \frac{\lambda}{(n-1)(\pi^T P^{(0)} \pi)(\mathbb{E}\Theta)^2} P^{(0)}. \quad (6)$$

To compare our results to the true labels, we will use normalized mutual information (NMI). One can think of the confusion matrix  $R$  as a bivariate probability distribution, and of its row and col-

umn sums  $R_{i+}$  and  $R_{+j}$  as the corresponding marginals. Then the NMI is defined by [27] as  $\text{NMI}(c, e) = -\sum_{i,j} R_{ij} \log \frac{R_{ij}}{R_{i+}R_{+j}} (\sum_{i,j} R_{ij} \log R_{ij})^{-1}$ , and is always a number between 0 and 1 (perfect match). It is useful to have a few benchmark values of NMI for reference: for example, for large  $n$ , matching 50%, 70%, and 90% of the labels correspond to values of NMI of approximately 0.12, 0.26, and 0.58, respectively.

All figures show the performance of the following methods:  $K$ -means clustering on 1- and 2-degrees (DC), spectral clustering (SC), spectral clustering with perturbations (SCP), unconditional pseudo-likelihood (UPL) initialized with either DC or SCP, and conditional pseudo-likelihood (CPL), with the same two initial values for labelings. The number of outer iterations for UPL and CPL is set to  $T = 20$ ;  $n$ ,  $\lambda$ ,  $\gamma$  and the number of replications  $N$  are specified in the figures.

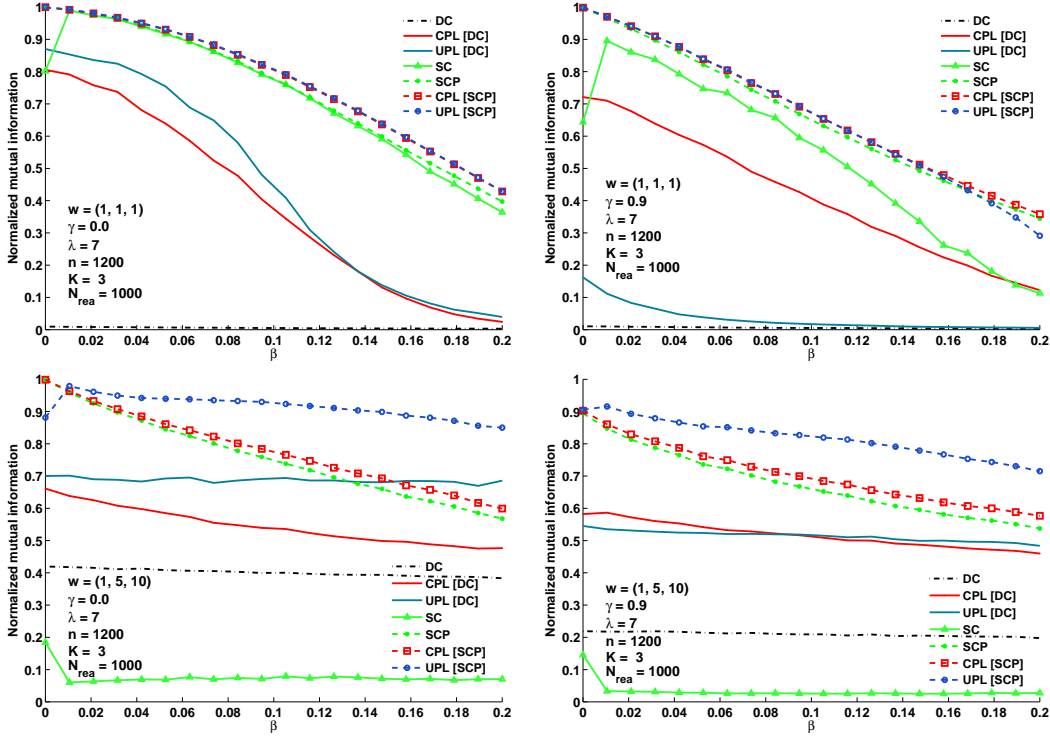


Figure 1: The NMI between true and estimated labels as a function of “out-in-ratio”  $\beta$ .

Figures 1 and 2 show results on estimating the node labels with varying  $\beta$  and  $\lambda$ , respectively. Generally, smaller  $\beta$  and larger  $\lambda$  make the problem easier, as we expect. In principle, degree-based clustering gives no information about the labels with uniform weights  $w$ , and only a moderate amount of information with non-uniform weights, so it serves as an example of a poor starting value for pseudo-likelihood. Regular spectral clustering performs well with uniform weights, but very poorly with non-uniform weights; we conjecture that this is due a limitation of  $K$ -means. Spectral clustering with perturbation, on the other hand, performs very well in all scenarios. Apart from being a useful general method on its own, it also serves as an example of a good starting value for pseudo-likelihood.

Figures 1 and 2 show that pseudo-likelihood achieves large gains over a poor starting value, giving surprisingly good results even when starting from the uninformative degree clustering in the case of  $w = (1, 1, 1)$ . One exception is unconditional pseudo-likelihood with  $\gamma = 0.9$  and  $w = (1, 1, 1)$ , which shows that conditioning is necessary to accomodate variation in degrees when the starting value is not very good. When spectral clustering with perturbation is used as a starting value, which is already very good, UPL and CPL do not have much room to do better, although UPL still provides a noticeable improvement, being overall the best method when initialized with SCP. It appears that a good starting value overcomes the limitations of the regular block model for networks with hubs, effectively ruling out the competing solution which divides nodes by degree.

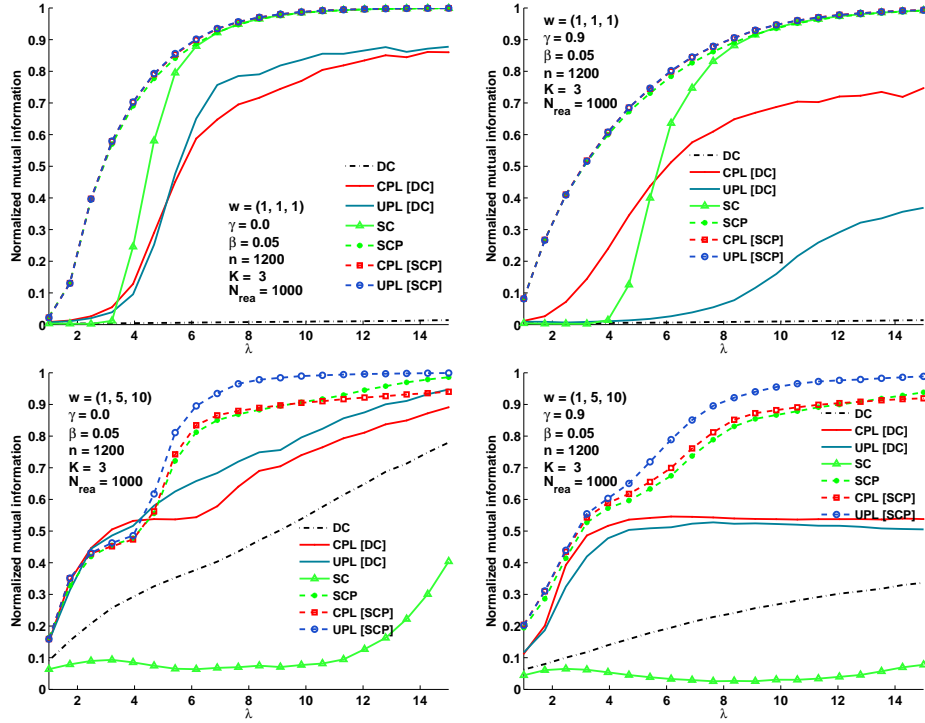


Figure 2: The NMI between true and estimated labels as a function of average expected degree  $\lambda$ .

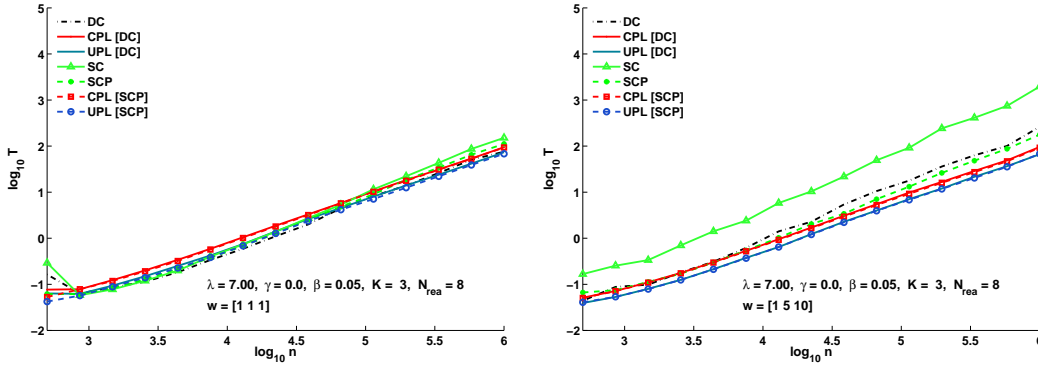


Figure 3: The runtime in seconds as a function of the number of nodes (log-log scale).

Finally, Figure 3 shows run times for all the methods for the case of the regular block model ( $\gamma = 0$ ) with different community weights ( $w = (1, 1, 1)$  and  $w = (1, 5, 10)$ ). The times shown for UPL and CPL do not include the time to compute the initial value, which is shown separately. For the case  $w = (1, 1, 1)$ , all methods take roughly the same amount of time. For the case  $w = (1, 5, 10)$ , spectral clustering (SC) takes considerably more time than the rest. On the other hand, SCP takes nearly the same time as it takes for  $w = (1, 1, 1)$ , and it slightly outperforms DC for larger values of  $n$ . This might be explained, in part, by the sparse matrix multiplication required for DC, which is both time and memory-consuming for large  $n$ . Generally, SCP provides an excellent starting value, with low computational complexity in a variety of situations.

## 4 Example: a political blogs network

This dataset on political blogs was compiled by [28] soon after the 2004 U.S. presidential election. The nodes are blogs focused on US politics and the edges are hyperlinks between these blogs. Each blog was manually labeled as liberal or conservative by [28], and we treat these as true community labels. Following [7], we ignore directions of the hyperlinks and analyze the largest connected component of this network, which has 1222 nodes and the average degree of 27. The distribution of degrees is highly skewed to the right (the median degree is 13, and the maximum is 351).

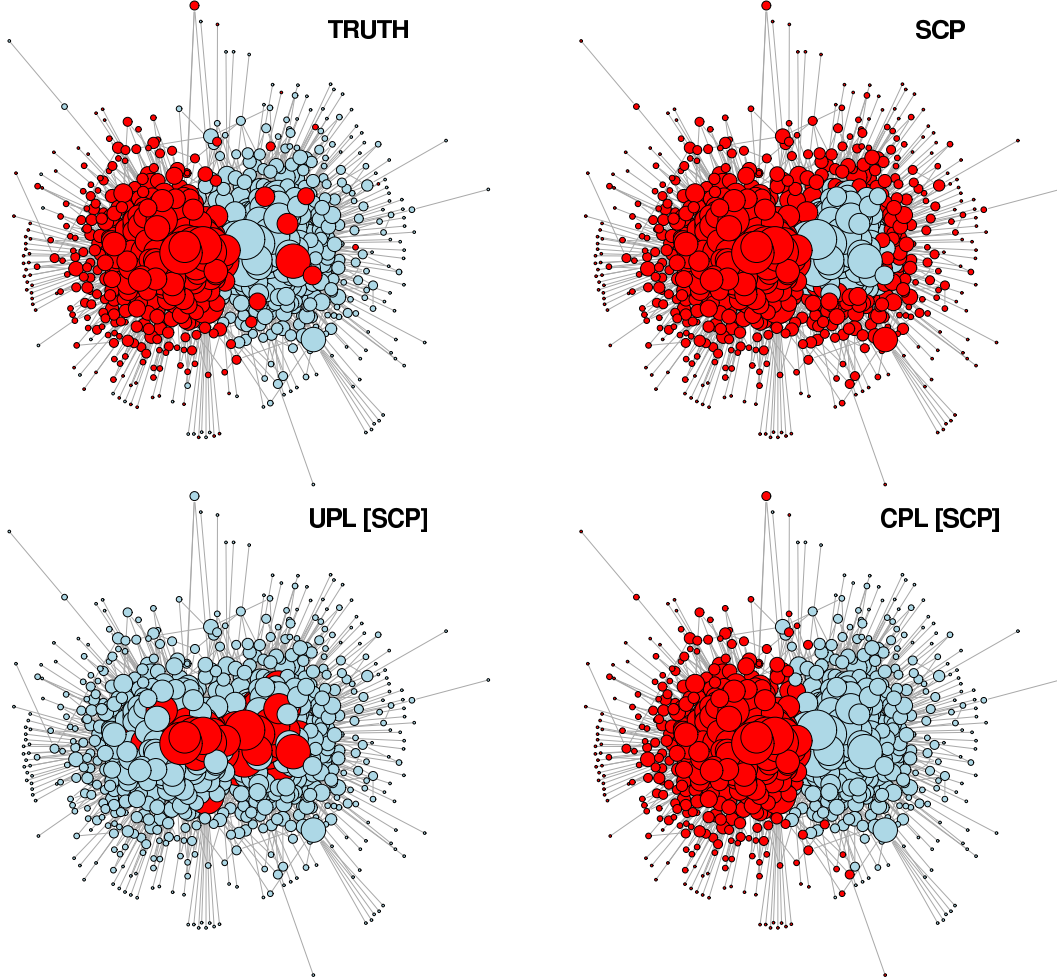


Figure 4: Political blogs data: true labels, spectral clustering with perturbations (SCP,  $\alpha = 0.25$ ), unconditional and conditional pseudo-likelihoods (UPL and CPL) initialized with SCP. Node size is proportional to log degree.

The results in Figure 4 show that the conditional pseudo-likelihood produces a result closest to the truth, as one would expect in view of highly variable degrees. Its result is also very close to those obtained by profile maximum likelihood for the degree-corrected block model and by two different modularities [7, 24]. Unconditional pseudo-likelihood, on the other hand, puts high-degree nodes in one group and low-degree nodes in the other. This is very close to the block model solution [7], which is a confirmation of unconditional pseudo-likelihood converging to the true block model estimate in this case. Spectral clustering with perturbations, with the default setting  $\alpha = 0.25$ , does something in between, misclassifying most of the low-degree nodes in one community (33% overall classification error), but yet provides a good enough starting value for CPL to recover a grouping close to the truth (5% classification error). Interestingly, a smaller  $\alpha$  used in SCP, e.g.,  $\alpha = 0.01$ , gives the same solution as CPL in this case. We hypothesize that this happens here because we only work with the largest connected component; dropping edges at random until the average degree was



5 and applying SCP with  $\alpha = 0.25$  to this much sparser network resulted in misclassifying only 6% of the nodes, confirming the hypothesis that the main advantage of SCP is correcting the problem caused by many small disconnected components. Regular spectral clustering does poorly, possibly because with  $K = 2$  communities we only keep one eigenvector and  $K$ -means in one dimension often has problems; if we keep a larger number of eigenvectors, say 5, spectral clustering performs as well as CPL.

## 5 Discussion

We have demonstrated empirically that the proposed algorithms provide fast and accurate community detection for a range of settings, including large and sparse networks. The pseudo-likelihood approximations we rely on have a long history of empirical success in statistics, but are not well studied theoretically. In ongoing work we are investigating the theoretical properties of the pseudo-likelihood algorithms as the expected degree  $\lambda \rightarrow \infty$  with  $n$ , as well as the theoretical properties of spectral clustering with perturbations. Developing a data-driven method for selecting the perturbation tuning parameter  $\alpha$  is also a topic for future work.

## References

- [1] M. E. J. Newman. Modularity and community structure in networks. *Proc. Natl. Acad. Sci. USA*, 103(23):8577–8582, 2006.
- [2] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75 – 174, 2010.
- [3] M. E. J. Newman. Detecting community structure in networks. *Eur. Phys. J. B*, 38:321–330, 2004.
- [4] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.
- [5] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69(2):026113, Feb 2004.
- [6] P. W. Holland, K. B. Laskey, and S. Leinhardt. Stochastic blockmodels: first steps. *Social Networks*, 5(2):109–137, 1983.
- [7] B. Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Physical Review E*, 83:016107, 2011.
- [8] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. *J. Machine Learning Research*, 9:1981–2014, 2008.
- [9] B. Ball, B. Karrer, and M. E. J. Newman. An efficient and principled method for detecting communities in networks. *Physical Review E*, 84:036103, 2011.
- [10] M. D. Handcock, A. E. Raftery, and J. M. Tantrum. Model-based clustering for social networks. *J. R. Statist. Soc. A*, 170:301–354, 2007.
- [11] P. D. Hoff. Modeling homophily and stochastic equivalence in symmetric relational data. In *Advances in Neural Information Processing Systems*, volume 19. MIT Press, Cambridge, MA, 2007.
- [12] P. O. Perry and P. J. Wolfe. Null models for network data. 2012. arXiv:1201.5871v1.
- [13] T. Snijders and K. Nowicki. Estimation and prediction for stochastic block-structures for graphs with latent block structure. *Journal of Classification*, 14:75–100, 1997.
- [14] K. Nowicki and T. A. B. Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455):1077–1087, 2001.
- [15] A. Celisse, J.-J. Daudin, and L. Pierre. Consistency of maximum-likelihood and variational estimators in the stochastic block model. 2011. arXiv:1105.3288.
- [16] M. Mariadassou, S. Robin, and C. Vacher. Uncovering latent structure in valued graphs: A variational approach. *The Annals of Applied Statistics*, 4(2):715–742, 2010.
- [17] P. J. Bickel and A. Chen. A nonparametric view of network models and Newman-Girvan and other modularities. *Proc. Natl. Acad. Sci. USA*, 106:21068–21073, 2009.
- [18] P. J. Bickel, A. Chen, and E. Levina. The method of moments for network models. *Ann. Statist.*, 2012. To appear.
- [19] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E*, 74(3):036104, Sep 2006.

- [20] K. Rohe, S. Chatterjee, and B. Yu. Spectral clustering and the high-dimensional stochastic block model. *Annals of Statistics*, 39(4):1878–1915, 2011.
- [21] J. E. Besag. Spatial interaction and the statistical analysis of lattice systems. *J. Roy. Statist. Soc., Ser. B*, 36:192–236, 1974.
- [22] M. E. J. Newman and E. A. Leicht. Mixture models and exploratory analysis in networks. *Proc. Natl. Acad. Sci. USA*, 104:9564–9569, 2007.
- [23] C. F. Jeff Wu. On the convergence properties of the EM algorithm. *The Annals of Statistics*, 11(1):95–103, 1983.
- [24] Y. Zhao, E. Levina, and J. Zhu. On consistency of community detection in network models. 2011. [arxiv.org/1110.3854](http://arxiv.org/1110.3854).
- [25] A. Channarond, J.-J. Daudin, and S. Robin. Classification and estimation in the stochastic block model based on the empirical degrees. 2011. [arxiv:1110.6517](http://arxiv.org/1110.6517).
- [26] A. Decelle, F. Krzakala, C. Moore, and L. Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, 84:066106, 2012.
- [27] Y. Y. Yao. Information-theoretic measures for knowledge discovery and data mining. In *Entropy Measures, Maximum Entropy Principle and Emerging Applications*, pages 115–136. Springer, 2003.
- [28] L. A. Adamic and N. Glance. The political blogosphere and the 2004 US election. In *Proceedings of the WWW-2005 Workshop on the Weblogging Ecosystem*, 2005.